AND'2010

Tokenizing Micro-Blogging Messages using a Text Classification Approach

Gustavo Laboreiro
Luís Sarmento
Jorge Teixeira
Eugénio Oliveira
LIACC - University of Porto / SAPO Labs - Portugal

Index

1. The tokenization problem
2. Tokenization as a classification problem
3. The features used
4. The testing scenarios
5. The experiment
6. Results
7. Improving the results
8. Conclusions and future work

Tokenization

- Separate a text into atomic units (words, numbers, punctuation, URLs, smileys, …)
- Tokenization quality has an impact in later text processing stages (error correction, normalization, semantic analysis, …)

The tokenization problem

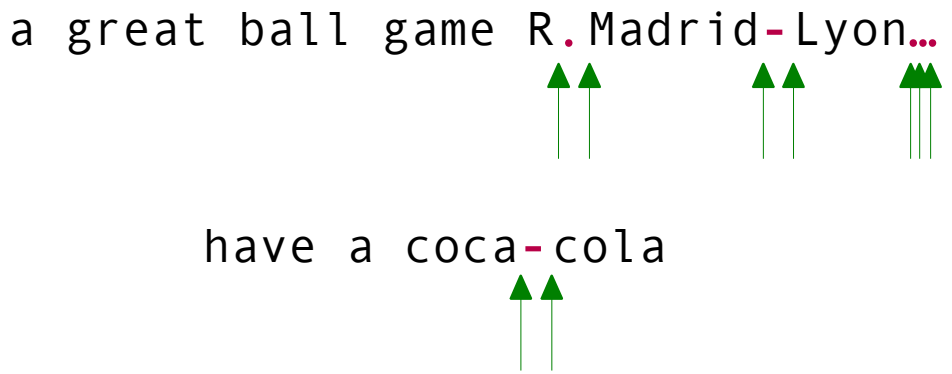
- *“well I'm going to watch what I think will be a great ball game R.Madrid-Lyon...see you soon dear twitterers...”*
  - *“(...) a great ball game R. Madrid - Lyon … (...)”*
- *What about these:*
  - *“@martiiinha don't you have an account at last.fm?”*
  - *“@carolisklassen Lets go and have a coca-cola and sleep on the table.”*

The tokenization problem

- User-Generated Content is particularly hard
  - *“@Neiazitahh is going?? O.O That is serious!! xD”*
  - *@VillegasBeliebr hey Taylor! Thanks :) <3*
  - *(loOoL) olo (THE DRA.MA OF THAT DUMMYy is he won'tT leave HOME ALONEe.) Drama. Drama. LtsS of DRA,MA.*
- There are numerous rules, “counter-rules” and exceptions
- We don't want to write them manually

Classification problem

- Insert or not a space character
- Decision points (*DP*)
  - Left or right of a non-alphanumeric character
  - Not next to a space

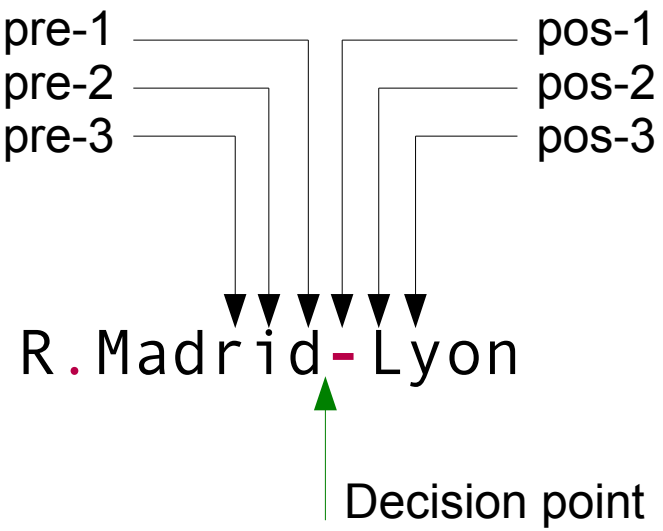


Features

- The features only describe characters
- Character nature
  - Alphabetic, numeric, symbol, space, ...
- Type of letter
  - Upper/lower case, accented, non-accented, vowel, ...
- Type of symbol
  - Bar, dash, monetary symbol, opening symbol, accent, arithmetic operator, smiley nose, ...

Features

- Feature character position
  - pre / post



- Feature window size
  - Limited due to vector space dimension
  - Typically, of size 10 or less
    (number of characters on each side of the *DP*)

Features

- The literal character feature
  - Handles cases where only certain sequences apply (e.g. “Mr.”, “Mrs.”, “Dr.”, “St.”, “http://”, “bit.ly”)
  - *is_literal_character_r_pre-1*
  - *is_literal_character_._pos-1*

Testing scenario S_{all}

- Remove all DP spaces from a test message
  - Can we put back the spaces?
  - Essentially, testing the recall
- One test / message:

Tokenized

a great ball game R. _Madrid_ - _Lyon_ ...

S_{all}

a great ball game R.Madrid-Lyon...

- The baseline is always inserting a space at the *DP*
  - There are many possible *DP* in this scenario

Testing scenario S_{one}

- Remove one DP space from the message
  - Can we put back the only missing space?
  - Essentially, testing the precision
- Several tests / message

Tokenized

```
a great ball game R._Madrid_-_Lyon_...
```

S_{one}

```
a great ball game R.Madrid - Lyon ...
a great ball game R. Madrid- Lyon ...
a great ball game R. Madrid -Lyon ...
  ...
```

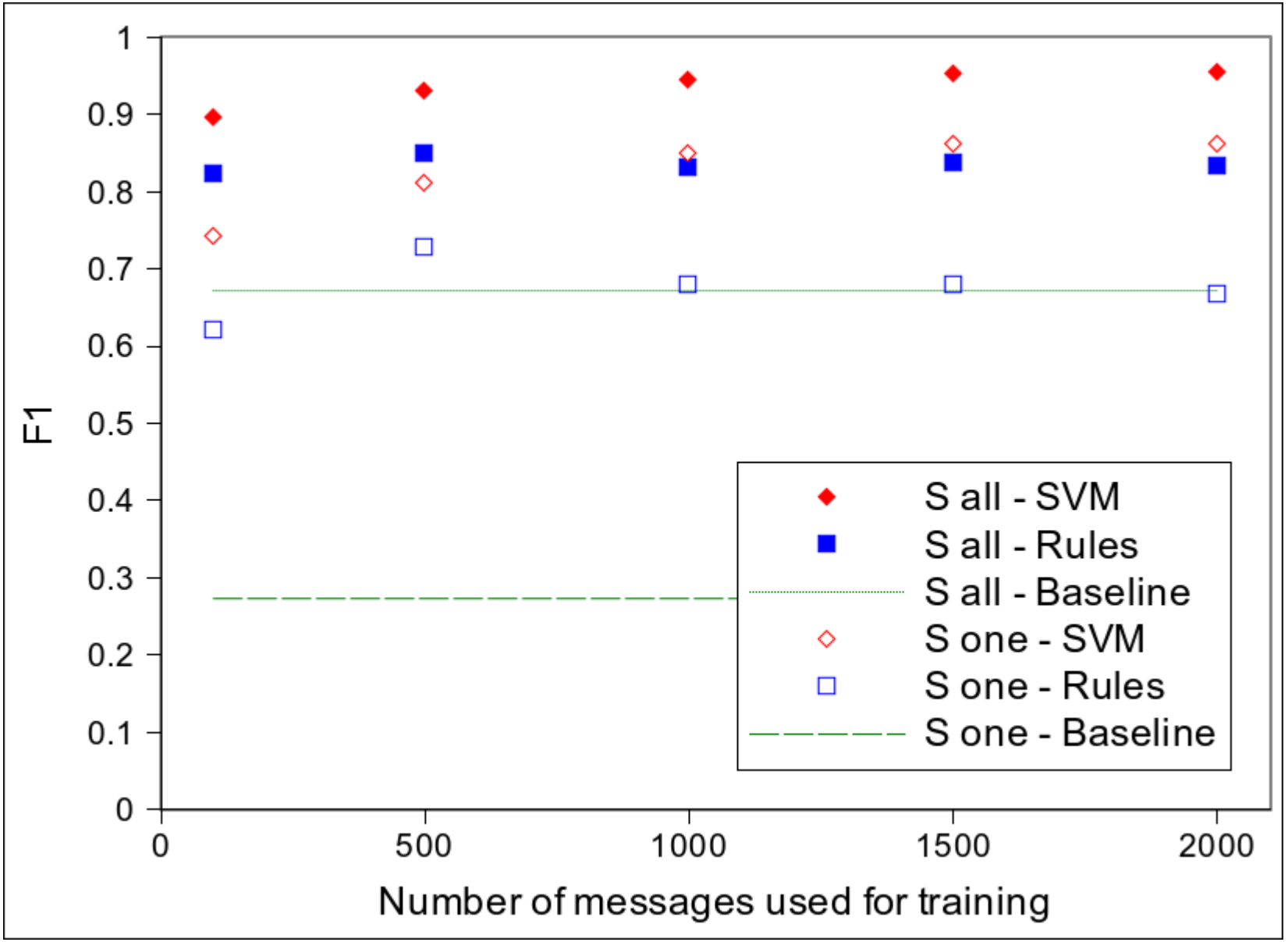
- The baseline is again always inserting a space at the DP
  - There are less decision points

The experiment

- Corpus: 2500 messages tokenized manually, mostly in *Portuguese*, but also includes *English*, *Spanish*, *French* and *others*
- Training: `R. Madrid - Lyon` → R. Madrid- Lyon / R. Madrid-Lyon
  - We used an SVM classifier
  - Bayesian classifier and ID3 were less satisfactory
- We looked at:
  - The impact of the training set
  - The impact of the feature window size
  - Using 5-fold Cross Validation

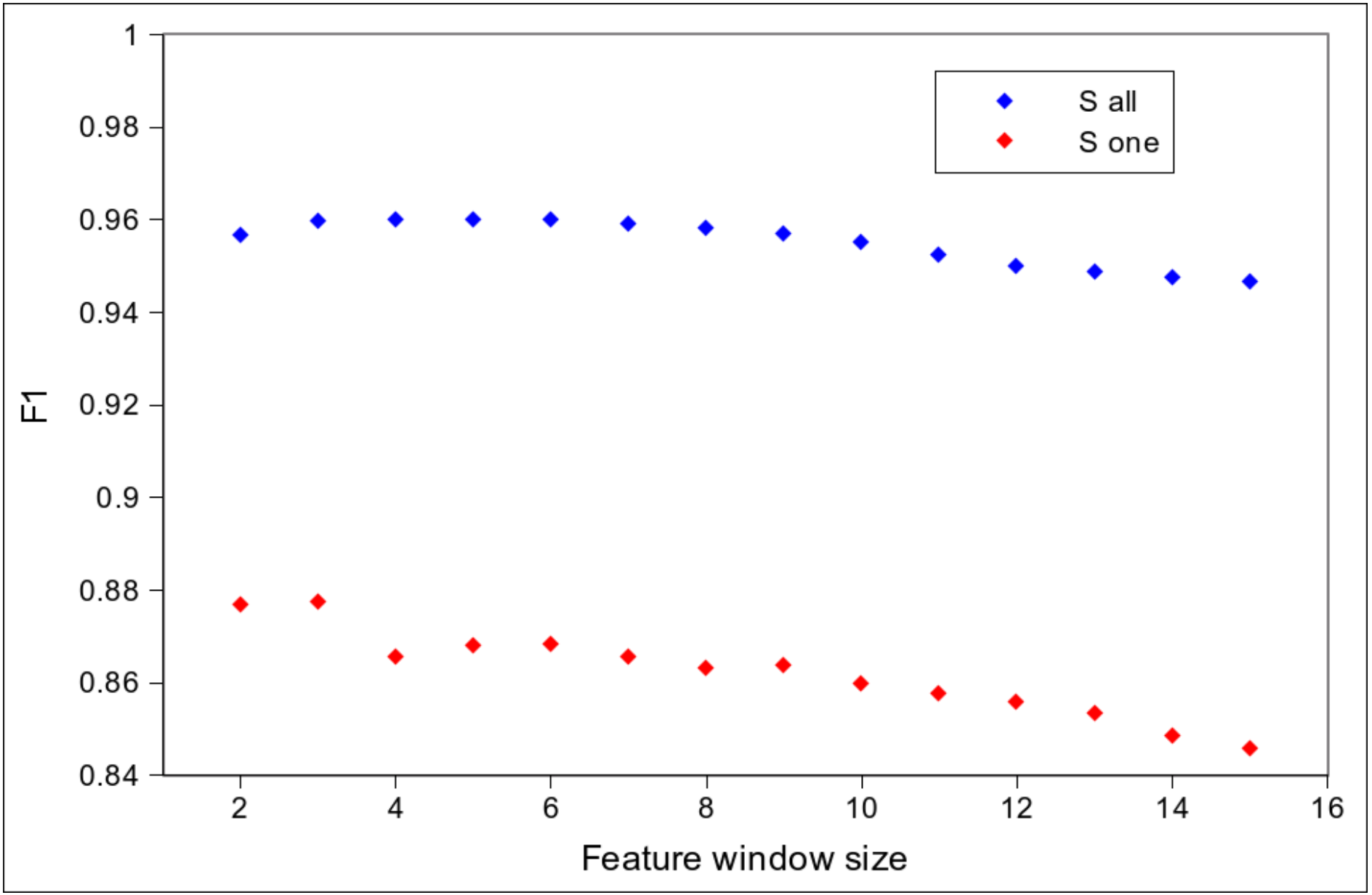
$$F1 = \frac{2PR}{P+R}$$

Training set size



$$F1 = \frac{2PR}{P+R}$$

Feature window size

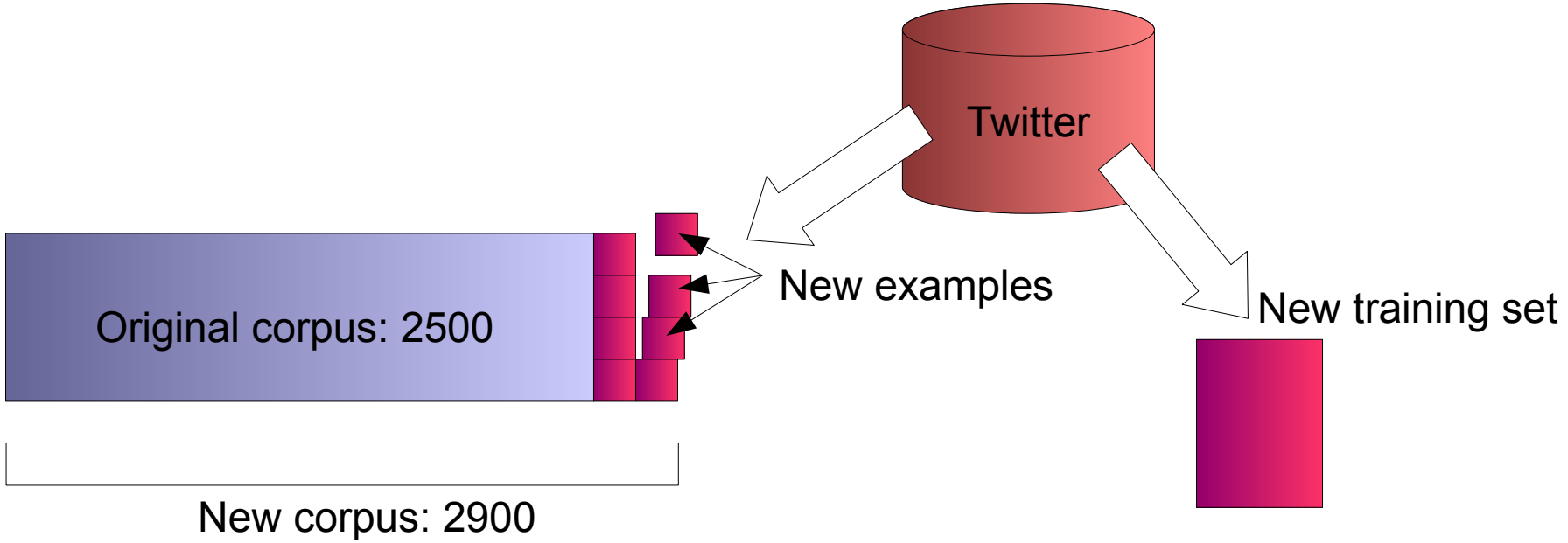


Problematic characters

Character	Errors	Msg. with occurr.
Dot .	1121	1587
Hyphen -	829	477
Apostrophe ' (very language dependent)	405	86
Colon :	350	1101
Slash /	172	725
Question mark ?	165	445
Opening parenthesis (	106	172
Closing parenthesis)	100	327

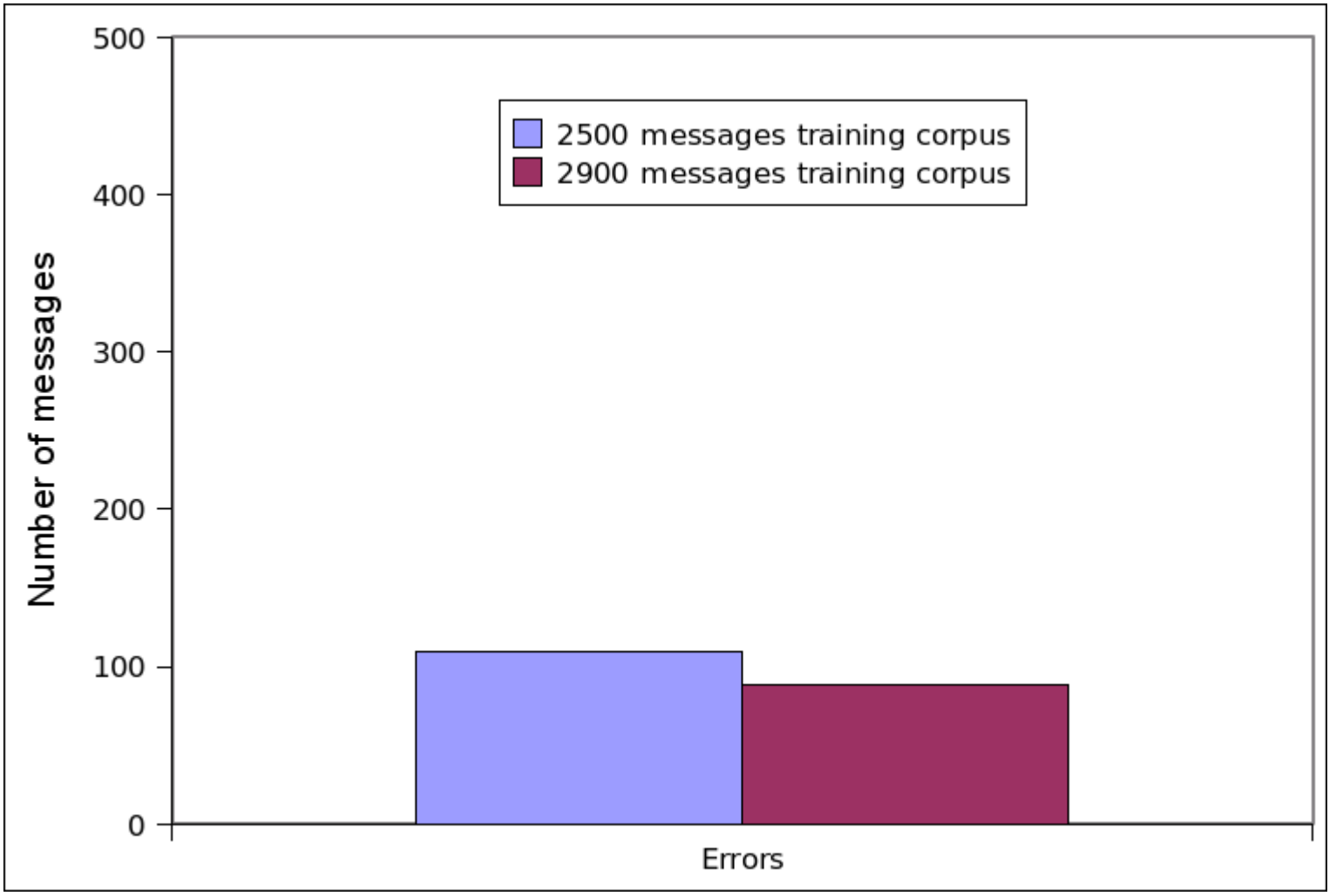
Adding more examples

- Added 400 new messages (50 / problematic character) to the corpus
- Tested the number of correctly tokenized messages against a new 500 messages corpus



Adding more examples

- Error rate reduced by 20%



Conclusions

- Text tokenization can be successfully expressed as a classification problem
- Learning approaches can easily outperform rule-based methods
- A small training set is enough, but larger is better
- A small feature window size is preferable to a large one
- Improving the results is simple and easy, compared to rules

Future work

- Explore SVM parameters
- Try other algorithms (CRF, for example)
- Language/user -dependent models
- Keyboard-related features